UNIT 3: NUMBER SYSTEM

3.1 Introduction of Decimal, Binary, Octal and Hexadecimal Number Systems.
3.2 Conversion of Decimal to Binary and Binary to Decimal
3.3 Binary addition & subtraction
3.4 ASCII and ANSI character code

TYPES OF NUMBER SYSTEM

There are various types of number systems in mathematics. The four most common number system types are:

1. Decimal number system (Base- 10)
2. Binary number system (Base- 2)
3. Octal number system (Base-8)
4. Hexadecimal number system (Base- 16)

DECIMAL NUMBER SYSTEM (BASE 10 NUMBER SYSTEM)

The decimal number system has a base of 10 because it uses ten digits from 0 to 9. In the decimal number system, the positions successive to the left of the decimal point represent units, tens, hundreds, thousands and so on. This system is expressed in decimal numbers. Every position shows a particular power of the base (10).

Example of Decimal Number System:

The decimal number 1457 consists of the digit 7 in the unit's position, 5 in the tens place, 4 in the hundreds position, and 1 in the thousands place whose value can be written as:

$(1\times10^3) + (4\times10^2) + (5\times10^1) + (7\times10^0)$

$(1\times1000) + (4\times100) + (5\times10) + (7\times1)$

$1000 + 400 + 50 + 7$

1457

BINARY NUMBER SYSTEM (BASE 2 NUMBER SYSTEM)

The base 2 number system is also known as the Binary number system wherein, only two binary digits exist, i.e., 0 and 1. Specifically, the usual base-2 is a radix of 2. The figures described under this system are known as binary numbers which are the combination of 0 and 1. For example, 110101 is a binary number.

OCTAL NUMBER SYSTEM (BASE 8 NUMBER SYSTEM)

In the octal number system, the base is 8 and it uses numbers from 0 to 7 to represent numbers. Octal numbers are commonly used in computer applications. Converting an octal number to decimal is the same as decimal conversion and is explained below using an example.

Example: 215

HEXADECIMAL NUMBER SYSTEM (BASE 16 NUMBER SYSTEM)

In the hexadecimal system, numbers are written or represented with base 16. In the hex system, the numbers are first represented just like in the decimal system, i.e. from 0 to 9. Then, the numbers are represented using the alphabet from A to F. The below-given table shows the representation of numbers in the hexadecimal number system.

Hexadecimal	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

CONVERSION

DECIMAL TO BINARY

Decimal numbers can be converted to binary by repeated division of the number by 2 while recording the remainder. Let’s take an example to see how this happens.

		Remainder	
2	43		
2	21	1	MSB
2	10	1	
2	5	0	
2	2	1	
2	1	0	
	0	1	LSB

The remainders are to be read from bottom to top to obtain the binary equivalent.

$43_{10} = 101011_2$

DECIMAL TO BINARY (FRACTION NUMBER)

Given a fraction decimal number n and integer k, convert decimal number n into equivalent binary number up-to k precision after decimal point.

Examples:

Input: n = 2.47, k = 5
Output: 10.01111

Input: n = 6.986 k = 8
Output: 110.11111100

We strongly recommend that you click here and practice it, before moving on to the solution.

A) Convert the integral part of decimal to binary equivalent
Divide the decimal number by 2 and store remainders in array.
Divide the quotient by 2.
Repeat step 2 until we get the quotient equal to zero.
Equivalent binary number would be reverse of all remainders of step 1.
B) Convert the fractional part of decimal to binary equivalent
Multiply the fractional decimal number by 2.
Integral part of resultant decimal number will be first digit of fraction binary number.
Repeat step 1 using only fractional part of decimal number and then step 2.
C) Combine both integral and fractional part of binary number.

Illustration:
Let's take an example for n = 4.47 k = 3

Step 1: Conversion of 4 to binary
1. 4/2 : Remainder = 0 : Quotient = 2
2. 2/2 : Remainder = 0 : Quotient = 1
3. 1/2 : Remainder = 1 : Quotient = 0

So equivalent binary of integral part of decimal is 100.

Step 2: Conversion of .47 to binary
1. 0.47 * 2 = 0.94, Integral part: 0
2. 0.94 * 2 = 1.88, Integral part: 1
3. 0.88 * 2 = 1.76, Integral part: 1

So equivalent binary of fractional part of decimal is .011

Step 3: Combined the result of step 1 and 2.

Final answer can be written as:
100 + .011 = 100.011

DECIMAL TO OCTAL

Decimal numbers can be converted to octal by repeated division of the number by 8 while recording the remainder. Let's take an example to see how this happens.

		Remainder	
8	473		
8	59	1	MSD
8	7	3	↑
	0	7	LSD

Reading the remainders from bottom to top,
$473_{10} = 731_8$

DECIMAL TO HEXADECIMAL

Decimal numbers can be converted to octal by repeated division of the number by 16 while recording the remainder. Let's take an example to see how this happens.

		Remainder	
16	423		
16	26	7	↑
16	1	A	
	0	1	

Reading the remainders from bottom to top we get,
$423_{10} = 1A7_{16}$

BINARY TO DECIMAL:

In this conversion, binary number to a decimal number, we use multiplication method, in such a way that, if a number with base n has to be converted into a number with base 10, then each digit of the given number is multiplied from MSB to LSB with reducing the power of the base. Let us understand this conversion with the help of an example.
Example 1. Convert $(1101)_2$ into a decimal number.
Solution: Given a binary number $(1101)_2$.
Now, multiplying each digit from MSB to LSB with reducing the power of the base number 2.
$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$= 8 + 4 + 0 + 1$
$= 13$

Therefore, $(1101)_2 = (13)_{10}$

BINARY TO DECIMAL (FRACTION NUMBER)

Examples:
Input: n = 110.101
Output: 6.625

Input: n = 101.1101
Output: 5.8125

We strongly recommend that you click here and practice it, before moving on to the solution.
Following are the steps of converting binary fractional to decimal.
A) Convert the integral part of binary to decimal equivalent
Multiply each digit separately from left side of radix point till the first digit by 20, 21, 22,… respectively.
Add all the result coming from step 1.
Equivalent integral decimal number would be the result obtained in step 2.
B) Convert the fractional part of binary to decimal equivalent
Divide each digit from right side of radix point till the end by 21, 22, 23, … respectively.
Add all the result coming from step 1.
Equivalent fractional decimal number would be the result obtained in step 2.
C) Add both integral and fractional part of decimal number.

Illustration
Let's take an example for n = 110.101

Step 1: Conversion of 110 to decimal
=> 1102 = (1*22) + (1*21) + (0*20)
=> 1102 = 4 + 2 + 0
=> 1102 = 6
So equivalent decimal of binary integral is 6.

Step 2: Conversion of .101 to decimal
=> 0.1012 = (1*1/2) + (0*1/22) + (1*1/23)
=> 0.1012 = 1*0.5 + 0*0.25 + 1*0.125
=> 0.1012 = 0.625
So equivalent decimal of binary fractional is 0.625

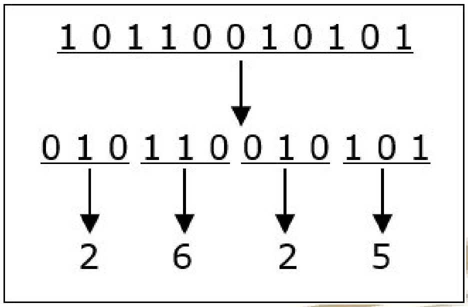
Step 3: Add result of step 1 and 2.
=> 6 + 0.625 = 6.625

BINARY TO OCTAL

To convert a binary number to octal number, these steps are followed −

- Starting from the least significant bit, make groups of three bits.
- If there are one or two bits less in making the groups, 0s can be added after the most significant bit
- Convert each group into its equivalent octal number

Let’s take an example to understand this.



$1011001010_{12} = 2625_8$

To convert an octal number to binary, each octal digit is converted to its 3-bit binary equivalent according to this table.

Octal Digit	0	1	2	3	4	5	6	7
Binary Equivalent	000	001	010	011	100	101	110	111

$54673_8 = 101100110111011_2$

BINARY TO HEXADECIMAL

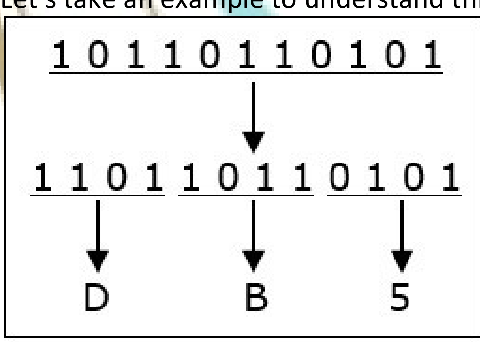
To convert a binary number to hexadecimal number, these steps are followed −

- Starting from the least significant bit, make groups of four bits.
- If there are one or two bits less in making the groups, 0s can be added after the most significant bit.
- Convert each group into its equivalent octal number.

Note:

Octal Digit	0	1	2	3	4	5	6	7
Binary Equivalent	0000	0001	0010	0011	0100	0101	0110	0111
Octal Digit	8	9	A(10)	B(11)	C(12)	D(13)	E(14)	F(15)
Binary Equivalent	1000	1001	1010	1011	1100	1101	1110	1111

Let’s take an example to understand this.



$10110110101_2 = DB5_{16}$

OCTAL TO DECIMAL:

To convert octal to decimal, we multiply the digits of octal number with decreasing power of the base number 8, starting from MSB to LSB and then add them all together.

Example 2: Convert 22_8 to decimal number.

Solution: Given, 22_8

$2 \times 8^1 + 2 \times 8^0$
= 16 + 2
= 18
Therefore, $22_8 = 18_{10}$

OCTAL TO BINARY

To convert octal to binary number, we can simply use the table. Just like having a table for hexadecimal and its equivalent binary, in the same way, we have a table for octal and its equivalent binary number.

Octal Number	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Example: Convert $(214)_8$ into a binary number.
Solution: From the table, we know,
2 → 010
1 → 001
4 → 100
Therefore, $(214)_8 = (010001100)_2$

OCTAL TO HEXADECIMAL

octal digit to 3 binary digits, with this table:

Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Then convert every 4 binary digits from bit0 to 1 hex digit, with this table:

Binary	Hex

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

EXAMPLE

Convert octal 154_8 to hex:

Convert every octal digit to 3 binary digits:

154_8 = 001 101 100 = 001101100_2

Then convert every 4 binary digits to 1 hex digit:

001101100_2 = 0110 1100 = $6C_{16}$

HEXADECIMAL TO BINARY

Convert every hex digit (start lowest digit) to 4 binary digits, with this table:

Hex	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

EXAMPLE

Convert hex $6C_{16}$ to binary:

$6C_{16}$ = 6 C = 110 1100 = 1101100_2

HEXADECIMAL TO OCTAL:

STEP 1: CONVERT $(FF)_{16}$ INTO BINARY

In order to convert the hexadecimal number into binary, we need to express every hexadecimal value using 4 binary bits.

Binary equivalent of **f** is $\mathbf{(1111)_2}$

= $(ff)_{16}$

= (1111)(1111)

= $(11111111)_2$

STEP 2: CONVERT $(11111111)_2$ INTO OCTAL

In order to convert the binary number into octal, we need to group every 3 binary bits and calculate the value [From left to right].

$(11111111)_2$ IN OCTAL

= $(11111111)_2$

= (11)(111)(111)

= $(377)_8$

HEXADECIMAL TO DECIMAL:

Example 3: Convert 121_{16} to decimal number.

Solution: $1 \times 16^2 + 2 \times 16^1 + 1 \times 16^0$

= 16 x 16 + 2 x 16 + 1 x 1

= 289

Therefore, $121_{16} = 289_{10}$

OPERATION ON NUMBER SYSTEM

ADDITION:

RULES OF BINARY ADDITION

Binary addition is much easier than the decimal addition when you remember the following tricks or rules. Using these rules, any binary number can be easily added. The four rules of binary addition are:

- 0 + 0 = 0
- 0 + 1 = 1
- 1 + 0 = 1
- 1 + 1 =10

HOW TO DO BINARY ADDITION?

Now, look at the example of the binary addition:101 + 101

Procedure for Binary Addition of Numbers:

    101
(+) 101

- **Step 1:** First consider the 1's column, and add the one's column,(1+1) and it gives the result 10 as per the condition of binary addition.
- **Step 2:** Now, leave the 0 in the one's column and carry the value 1 to the 10's column.

```
     1
    101
(+) 101
————–
      0
```

- **Step 3: Now add 10's place, 1+(0 + 0) = 1.** So, nothing carries to the 100's place and leave the value 1 in the 10's place

```
     1
    101
(+) 101
————-
     10
```

- **Step 4:** Now add the 100's place (1 + 1) = 10. Leave the value 0 in the 100's place and carries 1 to the 1000's place.

```
     1
    101
(+) 101
————-
   1010
```

So, the resultant of the addition operation is 1010.
When you cross-check the binary value with the decimal value, the resultant value should be the same.
The binary value 101 is equal to the decimal value 5
So, 5 + 5 = 10
The decimal number 10 is equal to the binary number 1010.

BINARY SUBTRACTION

Binary subtraction is the process of subtracting binary numbers. Binary numbers include only 0 and 1. The process of binary subtraction is the same as the arithmetic operation of subtraction that we do with numbers. Since only 0 and 1 are involved here, we may sometimes need to subtract 0 from 1. In such cases, we use the concept of borrowing as we do in an arithmetic subtraction. A binary number is expressed with a base-2. For example, a binary number is written as 10121012

RULES OF BINARY SUBTRACTION

There are some rules in which binary numbers are subtracted. They are,

$$\Rightarrow 1 - 0 = 1$$
$$\Rightarrow 1 - 1 = 0$$
$$\Rightarrow 0 - 0 = 0$$
$$\Rightarrow 0 - 1 = 1$$

(This can not be done directly, hence we borrow one digit from the digit to the left or the next higher order digit.)

HOW TO DO BINARY SUBTRACTION?

Decimal or base-10 numbers can be expressed as binary numbers. Binary numbers are used in computers to represent data since they understand only binary digits, 0 and 1. Let us understand how to subtract binary numbers with the example shown below.

Case i) - Binary subtraction without borrowing

Subtract 10021002 from 1111211112. Here number 4 is represented in binary as 10021002 and number 15 is represented as 1111211112.

Step 1: Arrange the numbers as shown in the figure below.

```
  1 1 1 1
-   1 0 0
_________
_________
```

Step 2: Follow the binary subtraction rules to subtract the numbers. In this subtraction, we do not encounter the subtraction of 1 from 0. Hence, the difference is 1011210112.

```
  1 1 1 1
-   1 0 0
_________
  1 0 1 1
```

Step 3: The decimal equivalent of 1011210112 is 11. Hence the difference is correct.

Case ii) Binary subtraction with borrowing

Subtract 10121012 from 1001210012. Here number 5 is represented in binary as 10121012 and number 9 is represented as 1001210012.

Step 1: Arrange the numbers as shown below.

```
  1 0 0 1
-   1 0 1
_________
_________
```

Step 2: Follow the binary subtraction rules to subtract the numbers. In this subtraction, first, let us subtract the numbers starting from the right and move to the next higher order digit. The first step is to subtract (1-1). This is equal to 0. Similarly, we move on to the next higher order digit and subtract (0 - 0), which is 0. In the next step, we have to subtract (0 - 1), so we borrow a 1 from the next higher order digit. Therefore, the result of subtracting (0 - 1) is 1.

```
  1 0 0 1
-   1 0 1
_________
    1 0 0
```

Step 3: Therefore, the difference of 1001210012 and 10121012 is 10021002. To verify this, let us find the decimal equivalent of 10021002, which is 4, Therefore, 9 - 5 = 4.

BINARY MULTIPLICATION

Binary multiplication is the process of multiplying binary numbers. The process of multiplying binary numbers is the same as that of arithmetic multiplication with decimal numbers. The only difference is that binary multiplication involves numbers that are consist of 0s and 1s, whereas, decimal multiplication involves numbers that comprise digits from 0 to 9. Let us learn the process of binary multiplication step by step.

BINARY MULTIPLICATION RULES

Binary multiplication is similar to the multiplication of decimal numbers. We have a multiplier and a multiplicand. The result of multiplication results in a product. Since only binary digits are involved in binary multiplication, we get to multiply only 0s and 1s. The rules for binary multiplication are as follows.

Multiplicand	**Multiplier**	**Product**
0	0	0 × 0 = 0
0	1	0 × 1 = 0
1	0	1 × 0 = 0
1	1	1 × 1 = 1

HOW TO MULTIPLY BINARY NUMBERS?

The process of multiplying binary numbers is similar and easier to do than decimal multiplication as binary numbers consist of only two digits which are 0 and 1. The method of multiplying binary numbers is given below. The same set of rules also apply to binary numbers with a decimal point. Let us take the example of multiplying (11101)211101)2 and (1001)21001)2. The decimal equivalent of (11101)211101)2 is 29 and the decimal equivalent of (1001)21001)2 is 9. Now let us multiply these numbers.

Step 1: Write down the multiplicand (11101)211101)2 and the multiplier (1001)21001)2 one below the other in proper positions.

Step 2: Multiply the rightmost digit or the least significant bit (LSB) of the multiplier (1) with all the digits of the multiplicand (11101)211101)2.

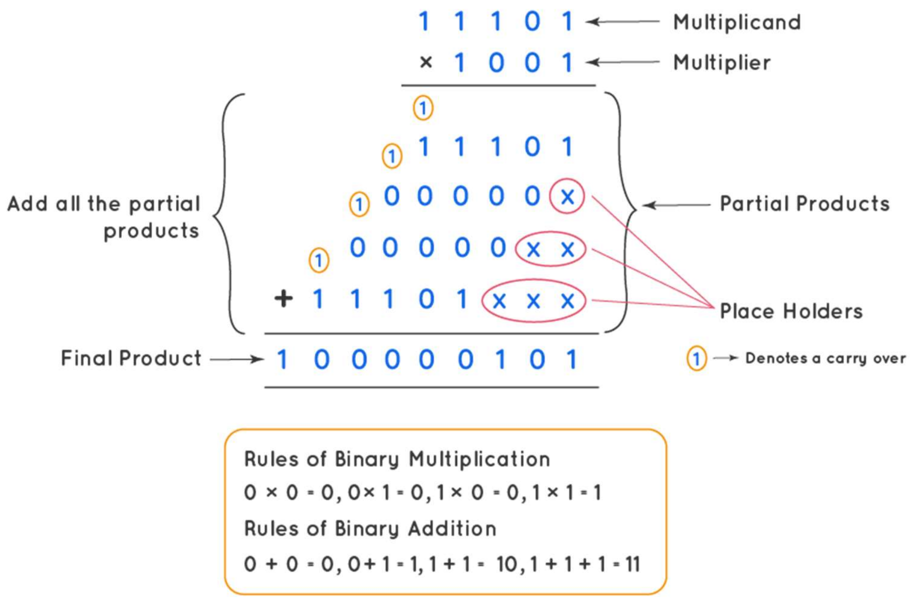
Step 3: Add a place holder of '0' or 'X' before multiplying the next higher order digit of the multiplier& with the multiplicand.

Step 4: Repeat the same process for all the next higher-order digits until we reach the most significant bit (MSB) which is the left-most digit of the multiplicand with the multiplier.

Step 5: The product obtained in each row is called the partial product. Finally, add all the partial products. To add all the binary numbers, use the rules of binary addition.

(The rules for binary addition are listed as follows: 0 + 0 = 0, 0 + 1 = 1, and 1 + 1 = 0, with a carryover of 1. So, 1 + 1 = 10 and 1 + 1 + 1 = 11 in the binary number system)

Let us look at the following process of binary multiplication as described above.



Therefore, the product of (11101)211101)2 and (1001)21001)2 is (100000101)2100000101)2. Let us verify our answer. The decimal equivalent of (100000101)2100000101)2 is 261. To know how to convert a binary number to a decimal number, click here. The decimal equivalent of& (11101)211101)2 is 29 and the decimal equivalent of (1001)21001)2 is 9. When we multiply 29 and 9 the product is 261. The decimal equivalent of (100000101)2100000101)2 is 261. Hence, the product is correct.

BINARY DIVISION

The binary division is one of the important operations of binary arithmetic. A binary number system or base-two is a counting technique that uses two digits: 0 and 1, and represents the number with the base 2. Here, the prefix 'bi' means 'two.' It is called binary as it has a base of 2 and it uses only two digits 0 and 1. Binary number systems are most commonly used in computer technology. All computers use a binary number system in their programming languages, that relies upon just two symbols, 0 and 1. With the rapid technological advancements across the globe, it is essential for one to understand the binary number system well.

In this article, we will learn the step-by-step method to make binary division easy to understand.

BINARY DIVISION

Binary division, similar to other binary arithmetic operations, is performed on binary numbers. The algorithm for binary division is somewhat similar to decimal division, the only difference here lies in the rules followed using the digits '0' and '1'. Binary multiplication and binary subtraction are the two binary arithmetic operations that are performed while performing binary division. The use of only '0' and '1' makes binary division quite simpler in comparison to decimal division. Other operations that are used while performing binary division are binary multiplication and binary subtraction.

BINARY DIVISION RULES

We can perform all arithmetic operations such as addition, subtraction, multiplication, and division on binary numbers, in the same way as we perform arithmetic operations on the decimal number system. Binary subtraction, binary multiplication, binary addition, and binary division are the four types of arithmetic operations that are performed here. We just need to follow some rules while dividing two binary numbers. There are four rules to be followed while performing binary division. Similar to the decimal system (or in any other number system), division by 0 is meaningless in Binary division. The binary division rules are as follows:

Dividend	Divisor	Result
0	1	0
1	1	1
Division by zero is meaningless		

The four rules given above are all the possible conditions for the division of binary numbers as binary numbers include only two digits 0 and 1.

HOW TO DO BINARY DIVISION?

Binary division problems can be solved by using the long division method, which is one of the most efficient and easiest ways to divide binary numbers. These are the steps to be followed in a binary division operation:

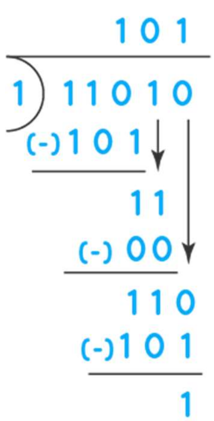
- **Step 1:** Compare the divisor with the dividend. If the divisor is larger, place 0 as the quotient, then bring the second bit of the dividend down. If the divisor is smaller, multiply it with 1 and the result becomes the subtrahend. Then, subtract the subtrahend from the minuend to get the remainder.
- **Step 2:** Then bring down the next number bit from the dividend portion and perform step 1 again.
- **Step 3:** Repeat the same process until the remainder becomes zero or the whole dividend is divided.

Let us understand binary division operation better using the following example:

Example: Consider two binary numbers, B = 01101020110102 and C = 0101201012 where we want to divide B by C.

Given: Dividend, 01101020110102 and the divisor = 0101201012.

Step1: Since the zero in the most significant bit position doesn't change the value of the number, let's remove it from both the dividend and divisor. So the dividend becomes 110102110102, and the divisor becomes 10121012.



Step 2: Let us use the long-division method. In this step, compare the divisor 10121012 with the first digit in the dividend 110102110102, since the divisor is smaller, it will be multiplied with 1 and the result will be the subtrahend.

As per the binary multiplication rules:

- 1 × 1 = 1
- 1 × 0 = 0
- 0 × 1 = 0
- 0 × 0 = 0

So, 101 × 1 = 10121012, and this result is written below.

Step 3: Subtract the subtrahend 10121012 from the minuend 11021102.

As per the binary subtraction rules,

- 0 - 1 = 1, we need to borrow 1 from the next more significant bit.
- 0 - 0 = 0
- 1 - 1 = 0

- 1 - 0 = 0

When we apply the above rules, this is how the calculation is done:

- For the first digit on the right, we have to subtract (0 - 1). So, we borrow a 1 from the digit on the left or the next higher order digit. Therefore, the result is 1.
- Then, (0 - 0 = 0) since the number in the next higher order digit becomes 0 after borrowing.
- 1 - 1 = 0 in the second next higher order digit.
- So, 11021102 - 10121012 = 00120012, and this result is written below.

Step 4: As per the rules of division, the next least significant bit comes down, and the divisor is multiplied by 1. Since the result, 10121012 is bigger than the minuend 0011200112, this step cannot be completed. Then, we have to go to the next step

Step 5: We write 0 as the next bit of the quotient and then, the least significant bit 0 comes down.

Step 6: Again the divisor is multiplied by 1 and the result is written as 101 × 1 = 10121012.

Step 7: Now we are at the final step. As per the binary subtraction, we subtract 10121012 from 11021102. We get, 11021102 - 10121012 = 00120012. The remainder is similar to Step 3, as all the numbers are the same. The binary division operation is completed now and we get the following result.

- Quotient = 10121012
- Remainder = 001 = 1

ASCII AND ANSI CHARACTER CODE

WHAT IS ANSI?

The American National Standards Institute (ANSI) codes are standardized numeric or alphabetic identifiers that are issued by the American National Standards Institute to enable uniform identification of geographic entities across all federal government departments. These codes can be found on ANSI documents. It is a generic term for the default code page of a given operating system, such as Windows

WHAT IS ASCII?

ASCII stands for the "American Standard Code for Information Interchange".
It was designed in the early 60's, as a standard character set for computers and electronic devices.
ASCII is a 7-bit character set containing 128 characters.
It contains the numbers from 0-9, the upper and lower case English letters from A to Z, and some special characters.

Basis of Comparison	ANSI	ASCII
Full Form	American National Standards Institute	American Standard Code for Information Interchange
Character Represents	256 characters	128 characters
Bit Uses	It uses 8-bit	It uses 7-bit
Life Span	It has shorter life span	It has longer life span
Consistency	Within the context of the system, these are not the same.	Every single system uses ASCII code points that are precisely the same.

Basis of Comparison	ANSI	ASCII
Complexity	It is not simple to use.	It is simple to use.
Standardized	No	Yes